

HW6

T1

-
- (a) 最后栈中剩: [A, F]
 - (b) 在 push d 或者 push e 后, 栈中元素最多
 - (c) 如果修改成队列后, 最后队列中剩: [E, F], 元素最多的时刻不变

T2

-
- 因为是中断服务, 所以 R7 不会保存返回地址, RET 会跳转到错误的位置
 - 直接调用 RET 没有恢复保存下来的 Processor Status Register, 因此程序没有从特权模式恢复到用户模式, 之前保存的状态码也没有恢复
 - RET 不涉及对栈的操作, 但从中断服务中恢复, 应该要恢复栈指针

T3

-
- (a) 程序可能反复读取同一个字符
 - (b) KBDR 存的数据可能还没有被读取, 用户输入字符可能丢失
 - (c) display hardware 并不会写入 DDR

T4

统计 x4000 开始的十个数据中首位为1 (负数) 的个数, 存在地址 x5000 处

T5

```
1      .ORIG x3000
2      LEA    R6, STACKBASE
3      LEA    R0, PROMPT
4      TRAP   x22
5      AND    R1, R1, #0
6      LOOP   TRAP x20
7      TRAP   x21
8      ADD    R3, R0, #-10
9      BRz   INPUTDONE
10     JSR    PUSH
11     ADD    R1, R1, #1
12     BRnzp LOOP
13     INPUTDONE ADD   R1, R1, #0
```

```

14          BRz    DONE
15  LOOP2      JSR    POP
16          TRAP   x21
17          ADD    R1,  R1,  #-1
18          BRp    LOOP2
19  DONE       TRAP   x25
20
21  PUSH       ADD    R6,  R6,  #-2
22          STR    R0,  R6,  #0
23          RET
24  POP        LDR    R0,  R6,  #-2
25          ADD    R6,  R6,  #2
26          RET
27  PROMPT     .STRINGZ "Please enter a sentence:"
28          STACKSPAC .BLKW #50
29          STACKBASE .FILL #0
30          .END

```

R1 寄存器的作用是记录输入字符串的长度，便于后续逆序输出

T6

```

1      .ORIG x3000
2      LD   R6, SP_INIT
3      LD   R0, LABELH
4      ADD R6, R6, #-1
5      STR R0, R6, #0
6      LD   R0, LABEL6
7      ADD R6, R6, #-1
8      STR R0, R6, #0
9      LD   R0, LABEL0
10     ADD R6, R6, #-1
11     STR R0, R6, #0
12     LD   R0, LABEL3
13     ADD R6, R6, #-1
14     STR R0, R6, #0
15     RET
16  LABELH .FILL x3012
17  LABEL6 .FILL x5018
18  LABEL0 .FILL x5000
19  LABEL3 .FILL x500c
20      HALT
21  SP_INIT .FILL xFE00
22          .END

```

代码的逻辑是把需要调用的子程序地址预先存入栈中，RET直接依次调用，注意 LABELH 的值是代码中 HALT 指令的地址，直接从 x3000 数行数得来

T7

- (a) 注意 $\times 10$ 是十进制的 16 所以循环次数是16次，开局三条指令+最后一次跳转： $8 \times 16 + 3 + 1 = 132$
(b) $144 \times 16 + 3 \times 27 + 16 = 2401$

指令	指令周期
LD	27
ADD	15
LDR	27
STR	26
BR (跳转)	16
BR (不跳转)	15

循环期间：1 个不跳转的 BR, 1个跳转的 BR, 4 个 ADD, 1 个 LDR, 1 个 SDR
则一次循环的指令数： $16 + 15 + 4 \times 15 + 27 + 26 = 144$